```
// <auto-generated />
using System;
using Microsoft.EntityFrameworkCore;
using Microsoft.EntityFrameworkCore.Infrastructure;
using Microsoft.EntityFrameworkCore.Metadata;
using Microsoft.EntityFrameworkCore.Migrations;
using Microsoft.EntityFrameworkCore.Storage.ValueConversion;
using MyCompany.Domain;

namespace MyCompany.Migrations
{
    [DbContext(typeof(AppDbContext))]
    [Migration("20200208061350__initial")]
    partial class _initial
    {
        protected override void BuildTargetModel(ModelBuilder modelBuilder)
        {
#pragma warning disable 612, 618
            modelBuilder
                .HasAnnotation("ProductVersion", "3.1.1")
                .HasAnnotation("Relational:MaxIdentifierLength", 128)
                .HasAnnotation("SqlServer:ValueGenerationStrategy",
SqlServerValueGenerationStrategy.IdentityColumn);

            modelBuilder.Entity("Microsoft.AspNetCore.Identity.IdentityRole", b =>
                {
                    b.Property<string>("Id")
                        .HasColumnType("nvarchar(450)");

                    b.Property<string>("ConcurrencyStamp")
                        .IsConcurrencyToken()
                        .HasColumnType("nvarchar(max)");

                    b.Property<string>("Name")
                        .HasColumnType("nvarchar(256)")
                        .HasMaxLength(256);

                    b.Property<string>("NormalizedName")
                        .HasColumnType("nvarchar(256)")
                        .HasMaxLength(256);

                    b.HasKey("Id");

                    b.HasIndex("NormalizedName")
                        .IsUnique()
                        .HasName("RoleNameIndex")
                        .HasFilter("[NormalizedName] IS NOT NULL");
```

```
            b.ToTable("AspNetRoles");

            b.HasData(
                new
                {
                    Id = "44546e06-8719-4ad8-b88a-f271ae9d6eab",
                    ConcurrencyStamp = "1c07fb23-ced7-48f9-bf0a-e0df233cd7a3",
                    Name = "admin",
                    NormalizedName = "ADMIN"
                });
        });

        modelBuilder.Entity("Microsoft.AspNetCore.Identity.IdentityRoleClaim<string>", b =>
        {
            b.Property<int>("Id")
                .ValueGeneratedOnAdd()
                .HasColumnType("int")
                .HasAnnotation("SqlServer:ValueGenerationStrategy",
SqlServerValueGenerationStrategy.IdentityColumn);

            b.Property<string>("ClaimType")
                .HasColumnType("nvarchar(max)");

            b.Property<string>("ClaimValue")
                .HasColumnType("nvarchar(max)");

            b.Property<string>("RoleId")
                .IsRequired()
                .HasColumnType("nvarchar(450)");

            b.HasKey("Id");

            b.HasIndex("RoleId");

            b.ToTable("AspNetRoleClaims");
        });

        modelBuilder.Entity("Microsoft.AspNetCore.Identity.IdentityUser", b =>
        {
            b.Property<string>("Id")
                .HasColumnType("nvarchar(450)");

            b.Property<int>("AccessFailedCount")
                .HasColumnType("int");

            b.Property<string>("ConcurrencyStamp")
                .IsConcurrencyToken()
                .HasColumnType("nvarchar(max)");
```

```
b.Property<string>("Email")
    .HasColumnType("nvarchar(256)")
    .HasMaxLength(256);

b.Property<bool>("EmailConfirmed")
    .HasColumnType("bit");

b.Property<bool>("LockoutEnabled")
    .HasColumnType("bit");

b.Property<DateTimeOffset?>("LockoutEnd")
    .HasColumnType("datetimeoffset");

b.Property<string>("NormalizedEmail")
    .HasColumnType("nvarchar(256)")
    .HasMaxLength(256);

b.Property<string>("NormalizedUserName")
    .HasColumnType("nvarchar(256)")
    .HasMaxLength(256);

b.Property<string>("PasswordHash")
    .HasColumnType("nvarchar(max)");

b.Property<string>("PhoneNumber")
    .HasColumnType("nvarchar(max)");

b.Property<bool>("PhoneNumberConfirmed")
    .HasColumnType("bit");

b.Property<string>("SecurityStamp")
    .HasColumnType("nvarchar(max)");

b.Property<bool>("TwoFactorEnabled")
    .HasColumnType("bit");

b.Property<string>("UserName")
    .HasColumnType("nvarchar(256)")
    .HasMaxLength(256);

b.HasKey("Id");

b.HasIndex("NormalizedEmail")
    .HasName("EmailIndex");

b.HasIndex("NormalizedUserName")
    .IsUnique()
```

```
                .HasName("UserNameIndex")
                .HasFilter("[NormalizedUserName] IS NOT NULL");

            b.ToTable("AspNetUsers");

            b.HasData(
                new
                {
                    Id = "3b62472e-4f66-49fa-a20f-e7685b9565d8",
                    AccessFailedCount = 0,
                    ConcurrencyStamp = "091be5bf-9d58-40e6-98a8-e82859894571",
                    Email = "my@email.com",
                    EmailConfirmed = true,
                    LockoutEnabled = false,
                    NormalizedEmail = "MY@EMAIL.COM",
                    NormalizedUserName = "ADMIN",
                    PasswordHash =
"AQAAAAEAACcQAAAAECBA6mt3xGNgyLjwyRhhtI3PI2IBKsm00Y3bAJfjdVrgT0++e45OV
5Vh4SLTLPDHEQ==",
                    PhoneNumberConfirmed = false,
                    SecurityStamp = "",
                    TwoFactorEnabled = false,
                    UserName = "admin"
                });
        });

        modelBuilder.Entity("Microsoft.AspNetCore.Identity.IdentityUserClaim<string>", b =>
            {
                b.Property<int>("Id")
                    .ValueGeneratedOnAdd()
                    .HasColumnType("int")
                    .HasAnnotation("SqlServer:ValueGenerationStrategy",
SqlServerValueGenerationStrategy.IdentityColumn);

                b.Property<string>("ClaimType")
                    .HasColumnType("nvarchar(max)");

                b.Property<string>("ClaimValue")
                    .HasColumnType("nvarchar(max)");

                b.Property<string>("UserId")
                    .IsRequired()
                    .HasColumnType("nvarchar(450)");

                b.HasKey("Id");

                b.HasIndex("UserId");
```

```
            b.ToTable("AspNetUserClaims");
        });

    modelBuilder.Entity("Microsoft.AspNetCore.Identity.IdentityUserLogin<string>", b =>
        {
            b.Property<string>("LoginProvider")
                .HasColumnType("nvarchar(450)");

            b.Property<string>("ProviderKey")
                .HasColumnType("nvarchar(450)");

            b.Property<string>("ProviderDisplayName")
                .HasColumnType("nvarchar(max)");

            b.Property<string>("UserId")
                .IsRequired()
                .HasColumnType("nvarchar(450)");

            b.HasKey("LoginProvider", "ProviderKey");

            b.HasIndex("UserId");

            b.ToTable("AspNetUserLogins");
        });

    modelBuilder.Entity("Microsoft.AspNetCore.Identity.IdentityUserRole<string>", b =>
        {
            b.Property<string>("UserId")
                .HasColumnType("nvarchar(450)");

            b.Property<string>("RoleId")
                .HasColumnType("nvarchar(450)");

            b.HasKey("UserId", "RoleId");

            b.HasIndex("RoleId");

            b.ToTable("AspNetUserRoles");

            b.HasData(
                new
                {
                    UserId = "3b62472e-4f66-49fa-a20f-e7685b9565d8",
                    RoleId = "44546e06-8719-4ad8-b88a-f271ae9d6eab"
                });
        });

    modelBuilder.Entity("Microsoft.AspNetCore.Identity.IdentityUserToken<string>", b =>
```

```
    {
        b.Property<string>("UserId")
            .HasColumnType("nvarchar(450)");

        b.Property<string>("LoginProvider")
            .HasColumnType("nvarchar(450)");

        b.Property<string>("Name")
            .HasColumnType("nvarchar(450)");

        b.Property<string>("Value")
            .HasColumnType("nvarchar(max)");

        b.HasKey("UserId", "LoginProvider", "Name");

        b.ToTable("AspNetUserTokens");
    });

modelBuilder.Entity("MyCompany.Domain.Entities.ServiceItem", b =>
    {
        b.Property<Guid>("Id")
            .ValueGeneratedOnAdd()
            .HasColumnType("uniqueidentifier");

        b.Property<DateTime>("DateAdded")
            .HasColumnType("datetime2");

        b.Property<string>("MetaDescription")
            .HasColumnType("nvarchar(max)");

        b.Property<string>("MetaKeywords")
            .HasColumnType("nvarchar(max)");

        b.Property<string>("MetaTitle")
            .HasColumnType("nvarchar(max)");

        b.Property<string>("Subtitle")
            .HasColumnType("nvarchar(max)");

        b.Property<string>("Text")
            .HasColumnType("nvarchar(max)");

        b.Property<string>("Title")
            .IsRequired()
            .HasColumnType("nvarchar(max)");

        b.Property<string>("TitleImagePath")
            .HasColumnType("nvarchar(max)");
```

```
            b.HasKey("Id");

            b.ToTable("ServiceItems");
        });

    modelBuilder.Entity("MyCompany.Domain.Entities.TextField", b =>
        {
            b.Property<Guid>("Id")
                .ValueGeneratedOnAdd()
                .HasColumnType("uniqueidentifier");

            b.Property<string>("CodeWord")
                .IsRequired()
                .HasColumnType("nvarchar(max)");

            b.Property<DateTime>("DateAdded")
                .HasColumnType("datetime2");

            b.Property<string>("MetaDescription")
                .HasColumnType("nvarchar(max)");

            b.Property<string>("MetaKeywords")
                .HasColumnType("nvarchar(max)");

            b.Property<string>("MetaTitle")
                .HasColumnType("nvarchar(max)");

            b.Property<string>("Subtitle")
                .HasColumnType("nvarchar(max)");

            b.Property<string>("Text")
                .HasColumnType("nvarchar(max)");

            b.Property<string>("Title")
                .HasColumnType("nvarchar(max)");

            b.Property<string>("TitleImagePath")
                .HasColumnType("nvarchar(max)");

            b.HasKey("Id");

            b.ToTable("TextFields");

            b.HasData(
                new
                {
                    Id = new Guid("63dc8fa6-07ae-4391-8916-e057f71239ce"),
```

```
                    CodeWord = "PageIndex",
                    DateAdded = new DateTime(2020, 2, 8, 6, 13, 50, 96,
DateTimeKind.Utc).AddTicks(9537),
                    Text = "Содержание заполняется администратором",
                    Title = "Главная"
                },
                new
                {
                    Id = new Guid("70bf165a-700a-4156-91c0-e83fce0a277f"),
                    CodeWord = "PageServices",
                    DateAdded = new DateTime(2020, 2, 8, 6, 13, 50, 97,
DateTimeKind.Utc).AddTicks(2218),
                    Text = "Содержание заполняется администратором",
                    Title = "Наши услуги"
                },
                new
                {
                    Id = new Guid("4aa76a4c-c59d-409a-84c1-06e6487a137a"),
                    CodeWord = "PageContacts",
                    DateAdded = new DateTime(2020, 2, 8, 6, 13, 50, 97,
DateTimeKind.Utc).AddTicks(2284),
                    Text = "Содержание заполняется администратором",
                    Title = "Контакты"
                });
        });

        modelBuilder.Entity("Microsoft.AspNetCore.Identity.IdentityRoleClaim<string>", b =>
        {
            b.HasOne("Microsoft.AspNetCore.Identity.IdentityRole", null)
                .WithMany()
                .HasForeignKey("RoleId")
                .OnDelete(DeleteBehavior.Cascade)
                .IsRequired();
        });

        modelBuilder.Entity("Microsoft.AspNetCore.Identity.IdentityUserClaim<string>", b =>
        {
            b.HasOne("Microsoft.AspNetCore.Identity.IdentityUser", null)
                .WithMany()
                .HasForeignKey("UserId")
                .OnDelete(DeleteBehavior.Cascade)
                .IsRequired();
        });

        modelBuilder.Entity("Microsoft.AspNetCore.Identity.IdentityUserLogin<string>", b =>
        {
            b.HasOne("Microsoft.AspNetCore.Identity.IdentityUser", null)
                .WithMany()
```

```
                    .HasForeignKey("UserId")
                    .OnDelete(DeleteBehavior.Cascade)
                    .IsRequired();
                });

            modelBuilder.Entity("Microsoft.AspNetCore.Identity.IdentityUserRole<string>", b =>
                {
                    b.HasOne("Microsoft.AspNetCore.Identity.IdentityRole", null)
                        .WithMany()
                        .HasForeignKey("RoleId")
                        .OnDelete(DeleteBehavior.Cascade)
                        .IsRequired();

                    b.HasOne("Microsoft.AspNetCore.Identity.IdentityUser", null)
                        .WithMany()
                        .HasForeignKey("UserId")
                        .OnDelete(DeleteBehavior.Cascade)
                        .IsRequired();
                });

            modelBuilder.Entity("Microsoft.AspNetCore.Identity.IdentityUserToken<string>", b =>
                {
                    b.HasOne("Microsoft.AspNetCore.Identity.IdentityUser", null)
                        .WithMany()
                        .HasForeignKey("UserId")
                        .OnDelete(DeleteBehavior.Cascade)
                        .IsRequired();
                });
#pragma warning restore 612, 618
        }
    }
}
```